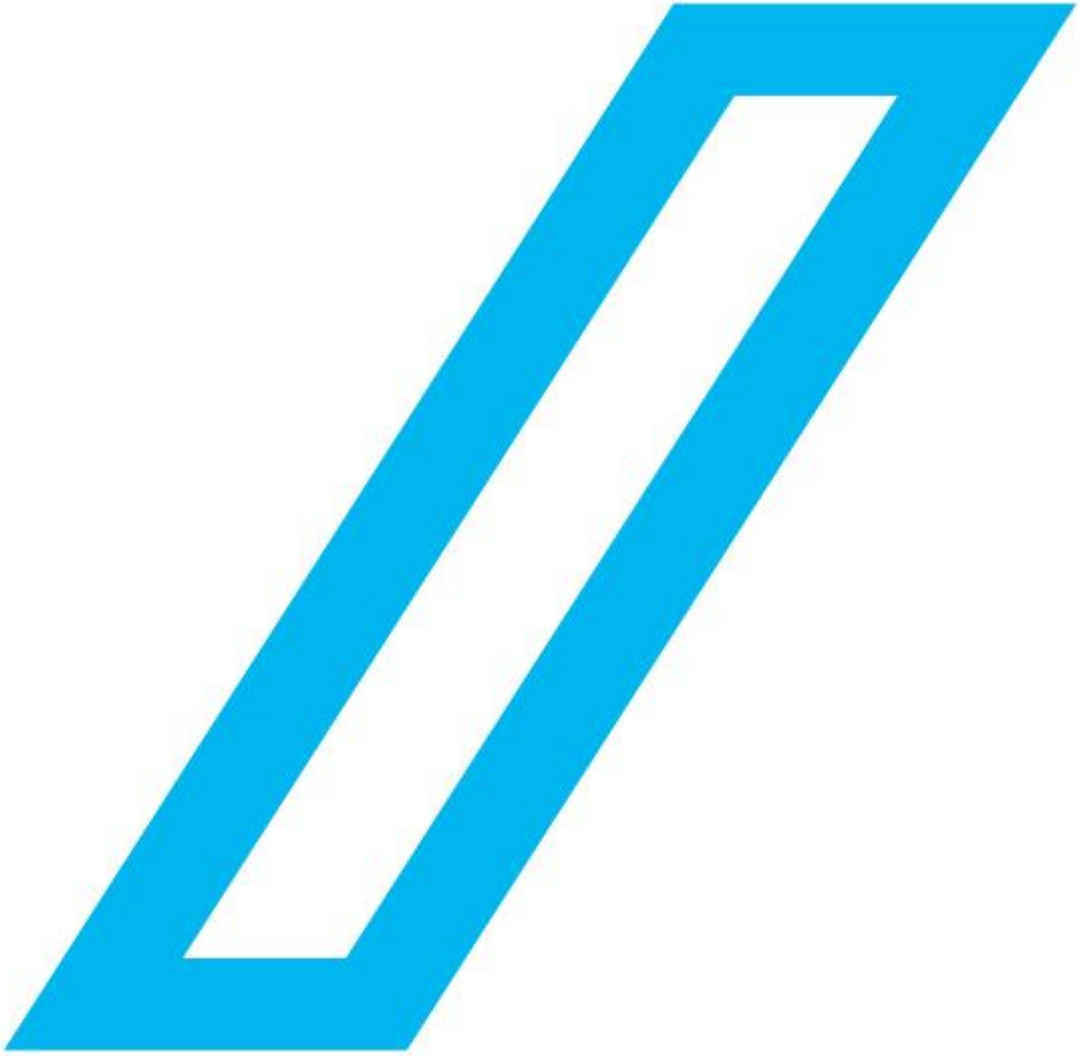
HL Klemove

PID Control

Lecturer : Seungmok Song



Contents

1. Introduction
2. P controller
3. PD controller
4. PID controller
5. Beyond PID controller

제어: 원하는 값을 갖도록 하는 것

완벽한 제어 = 완벽한 모델 ⊕ 완벽한 외란

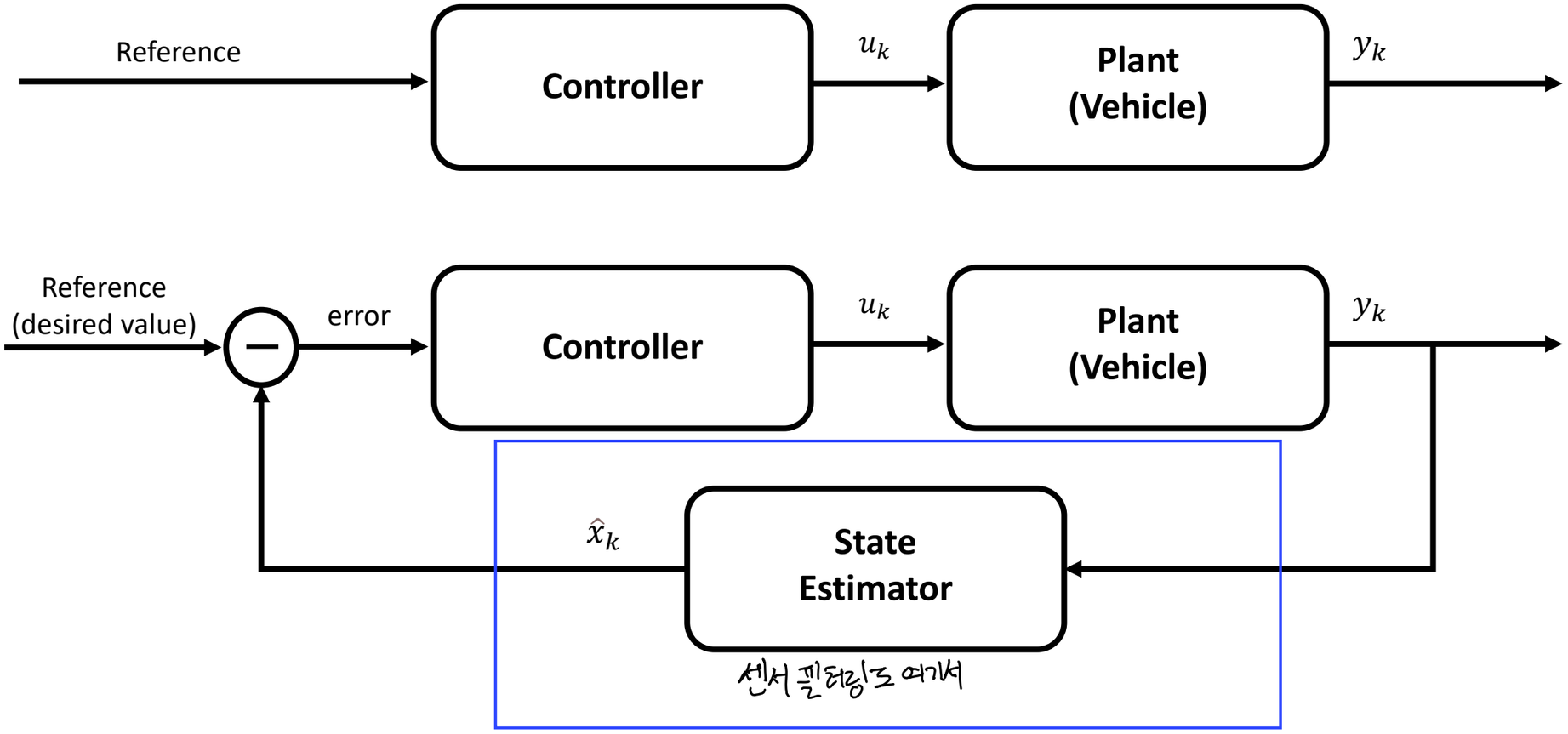
Introduction

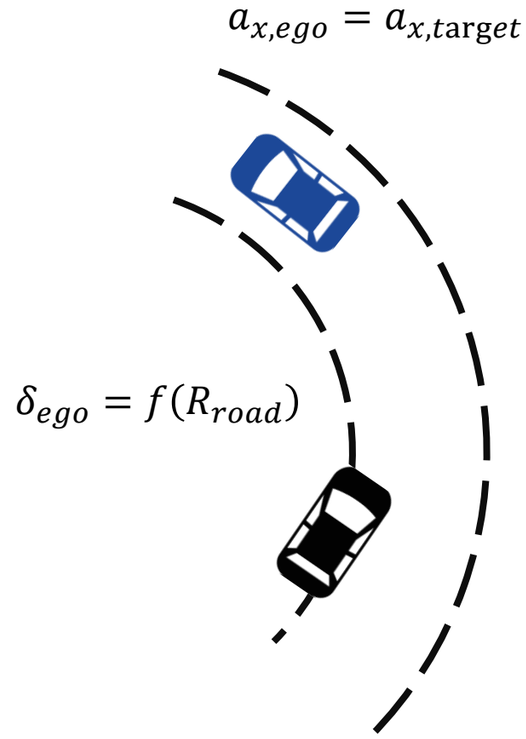
피드백 유무의 차이

- Feedback vs. Feedforward controller

~> 제어기는 이렇게 오기가 대부분.

↳ 모델링이 완벽할 때. 초기값 안으로 ~> 도저히 들어버면 절대 추종 불가능 애초 당연히 든다.

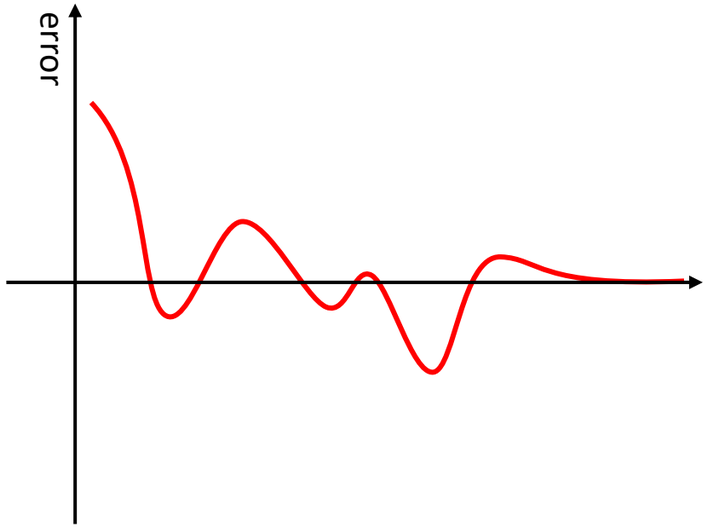


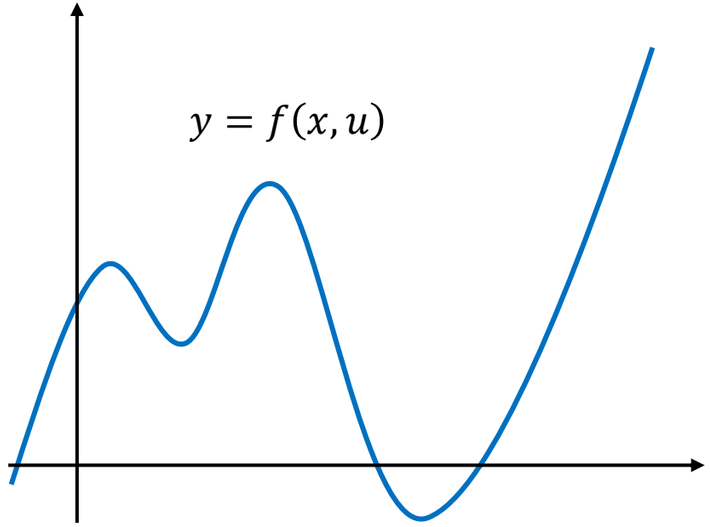


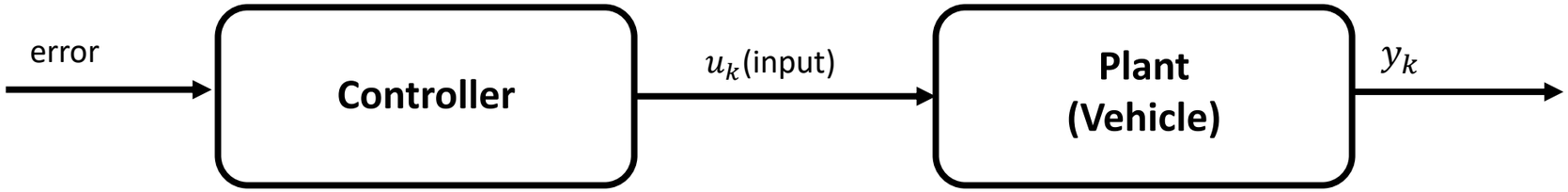
Introduction

- PID Controller
  - We don't have to care about the system model

~> 모델이 이면 비선형 복잡해져도 제어 가능
애초에 모델이 안 들어가니까.

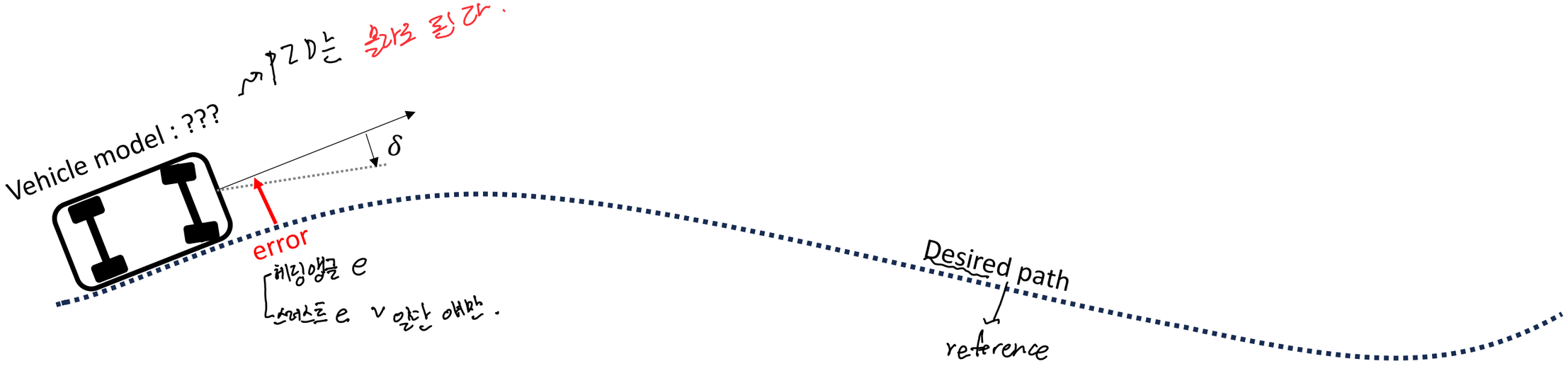






Introduction

- Path tracking example

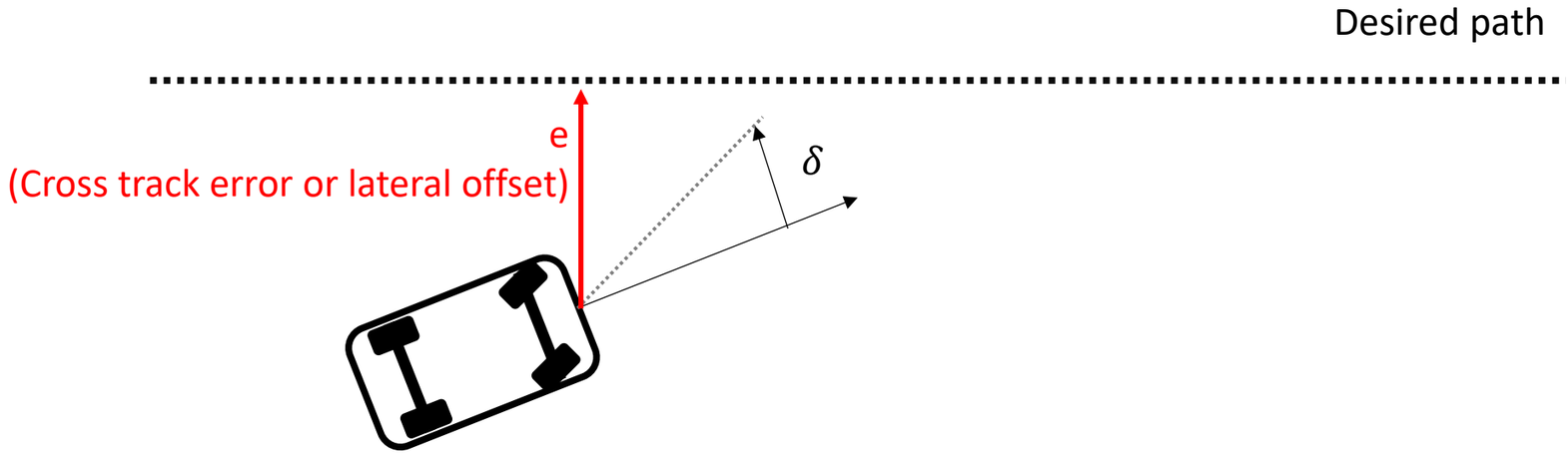


P Controller

- Error 에 비례한 control input 생성

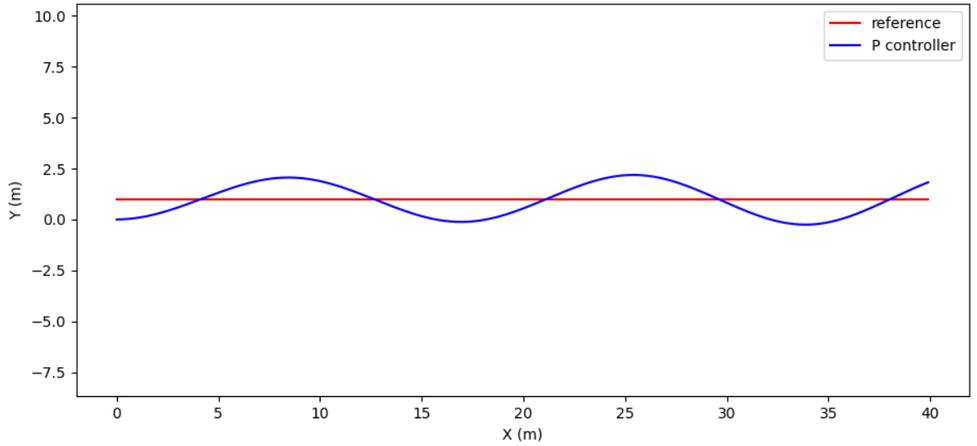
$$input(\delta) \propto error(e)$$

$$\delta_p = K_p e$$

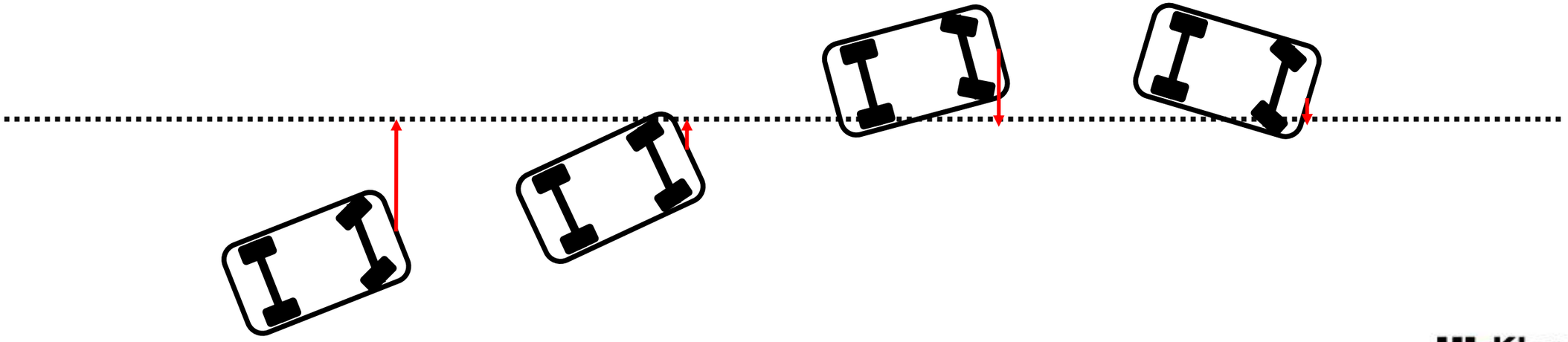


P Controller

- Result
  - Overshoot



→ 너무 간편한 피드백
→ overshoot
⇒ 이래서 PD제어기 쓴다.



1. P Controller

아래 그림과 같이 모형 자동차가 주어진 경로를 따라 주행하고자 합니다. 모형차는 Y 방향으로 힘 Fy를 가하여 제어가 가능합니다.

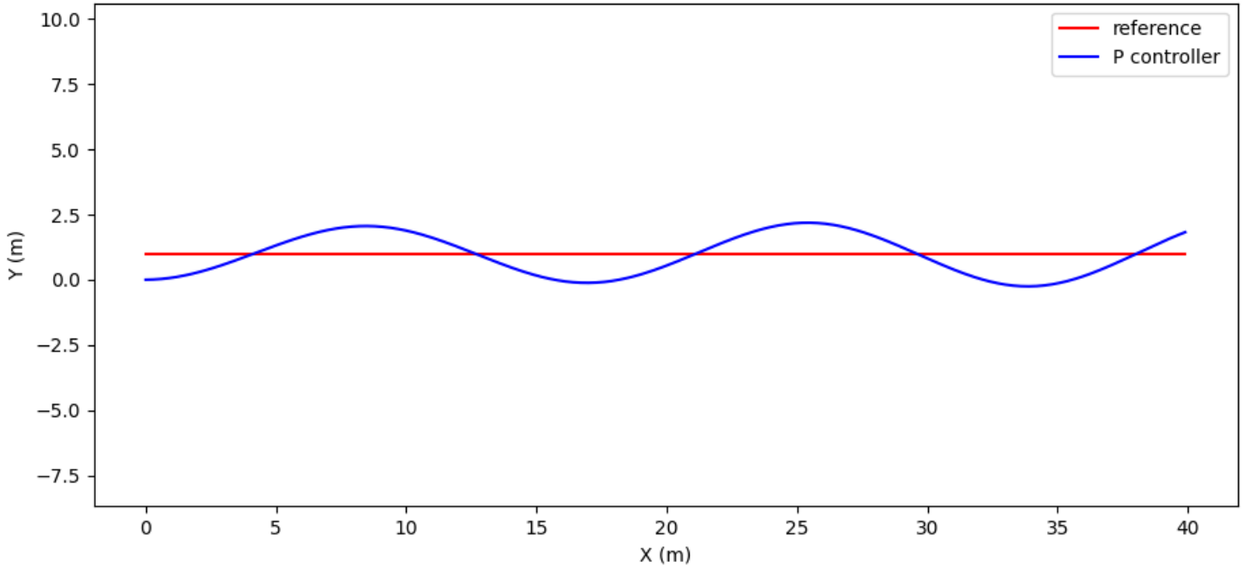
Error 에 비례한 제어를 하는 P 제어기를 설계하여 ex01_P_Controller.py 에 코드를 작성해 보세요. [10점]

control input = F_y

PD Controller

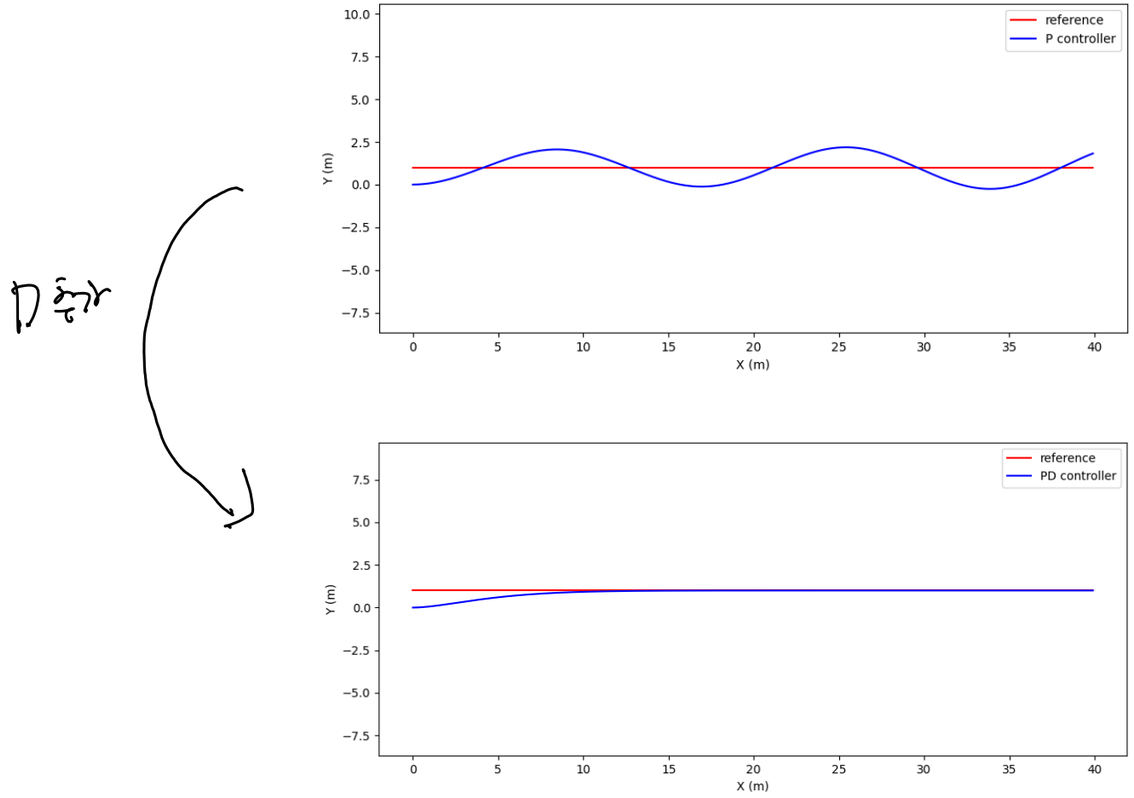
- Derivative term
  - Reducing overshoot

$$\delta_{pd} = K_p e + K_d \frac{d}{dt}(e)$$



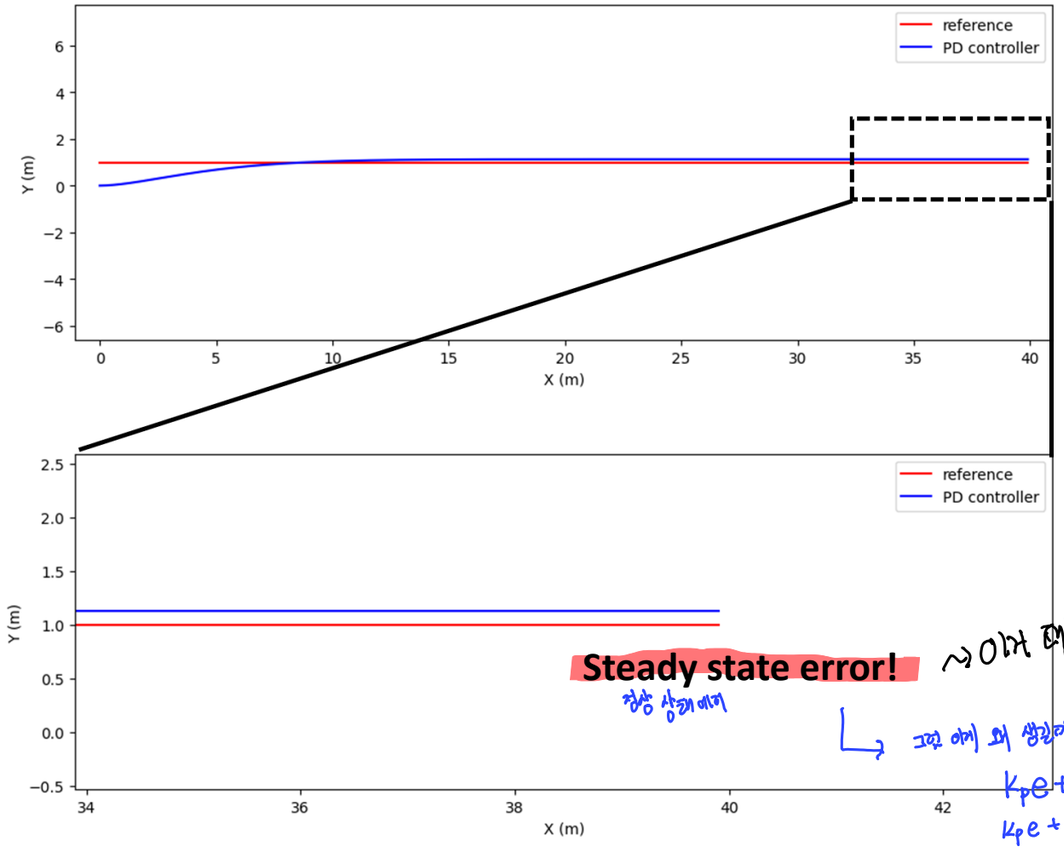
PD Controller

- Result



PD Controller

- Result
  - Bias



Steady state error!

정상 상태 에러

↳ 이거 해결에 I 제어기 등장

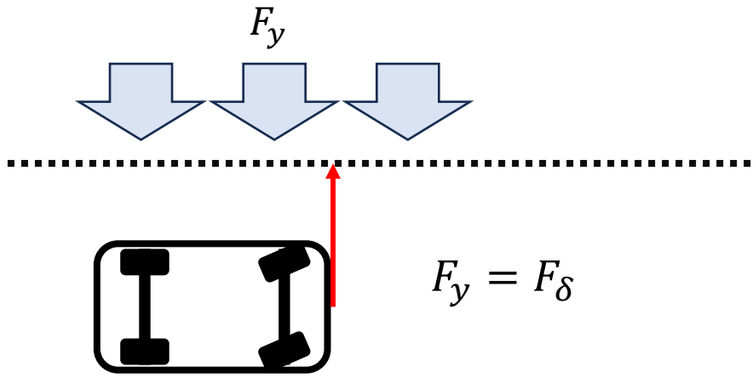
↳ 그럼 여기 왜 생길까?

$K_p e + K_d \dot{e}$

$K_p e + K_d \frac{de}{dt}$

→ 이론 상 없어지는게 맞음.

근데 현실 세계라 생기는거임.



$$\delta_{pd} = K_p e + K_d \frac{d}{dt}(e)$$

= 0

만약 타이어 공기압 한쪽이 높다고 생각해보자. 그럼 에러는 아무리 조정해도 존재할 수밖에 없음

↳ 이건 PD에서 제어 불가능.

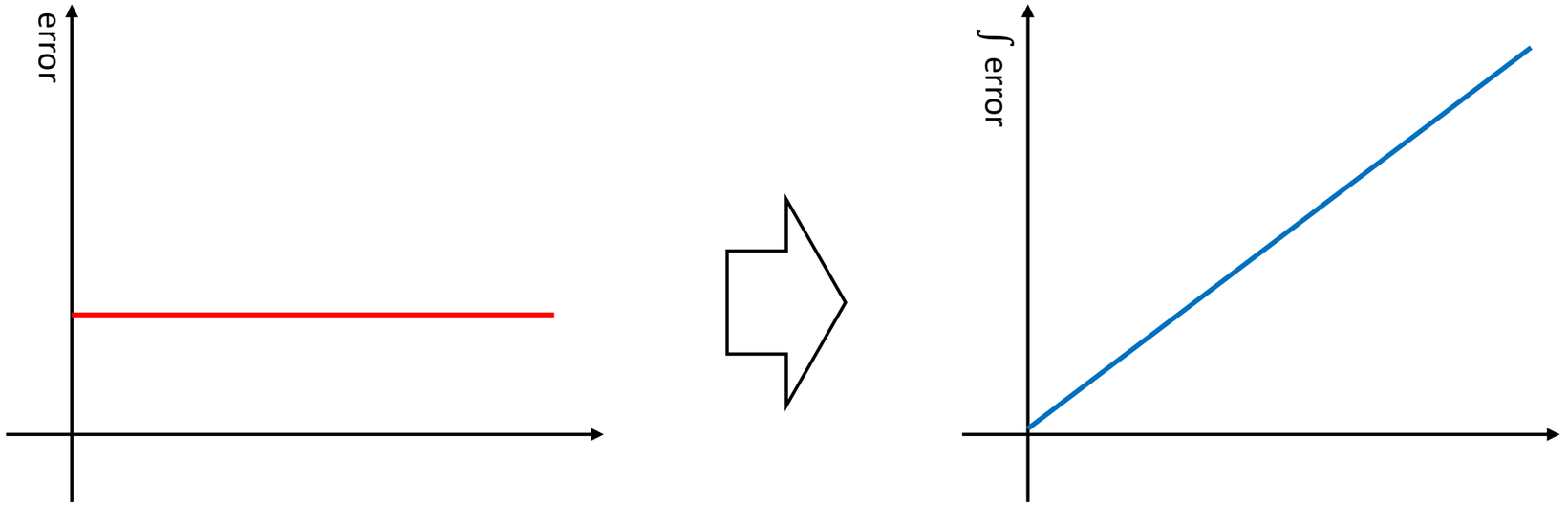
(일정한 외란)

PID Controller

- Integral term
  - Reducing steady state error

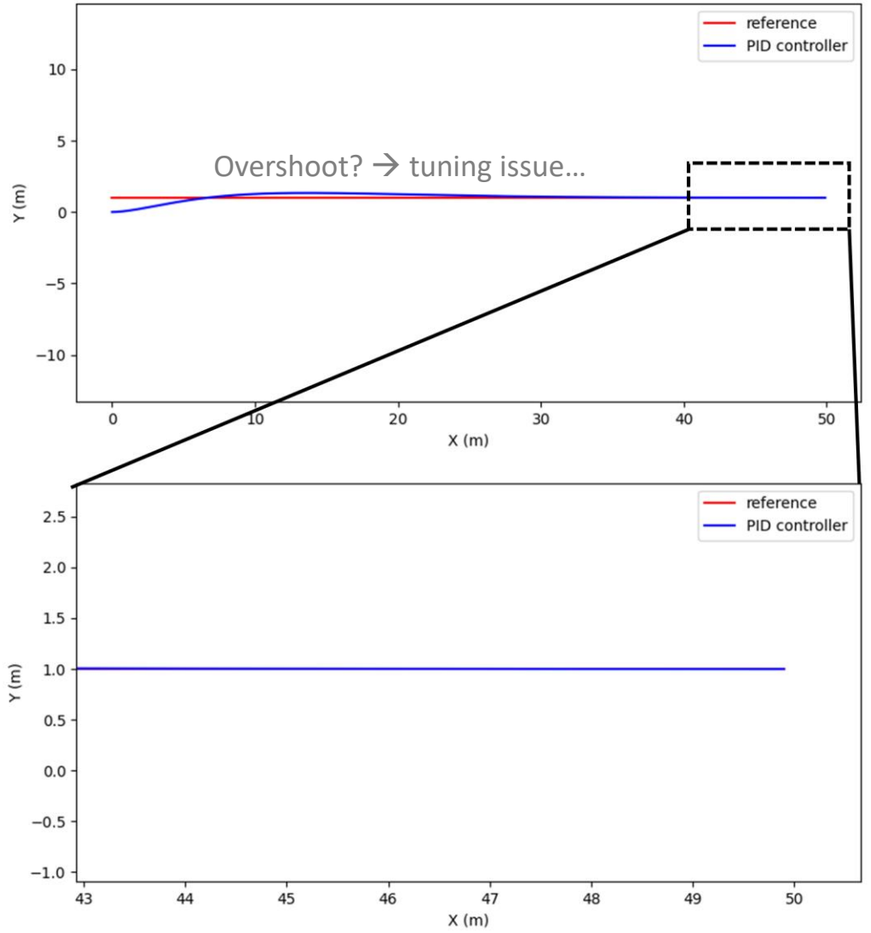
$$\delta_{pd} = K_p e + K_d \frac{d}{dt}(e) + K_i \int (e)dt$$

Error 가 존재하는 시간이 길어질 수록 커짐!



PID Controller

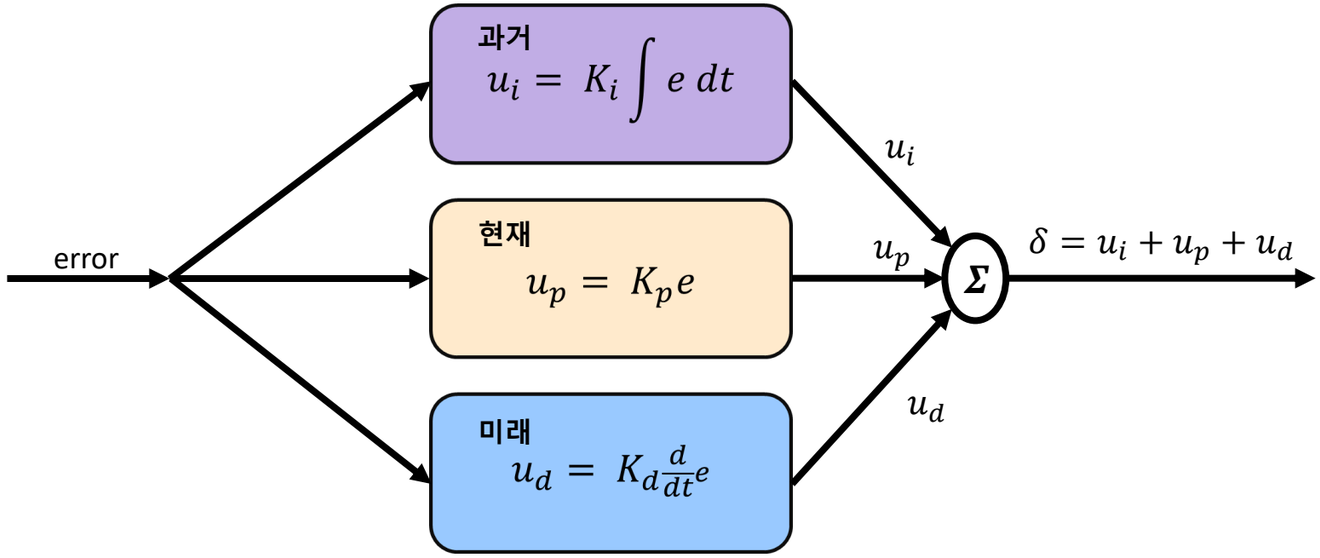
- Result

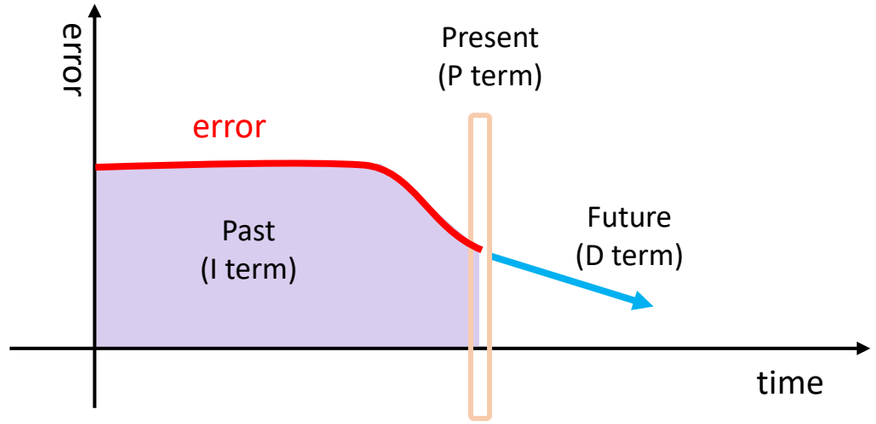


PID Controller

- Meanings of each controller

관약 그저께가 힘들면 그저께는 과감히 빼라.
난이도↑

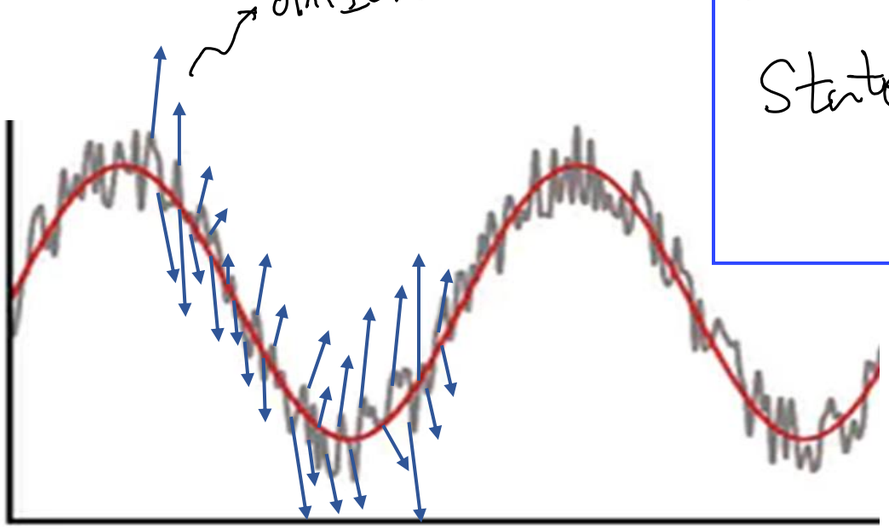




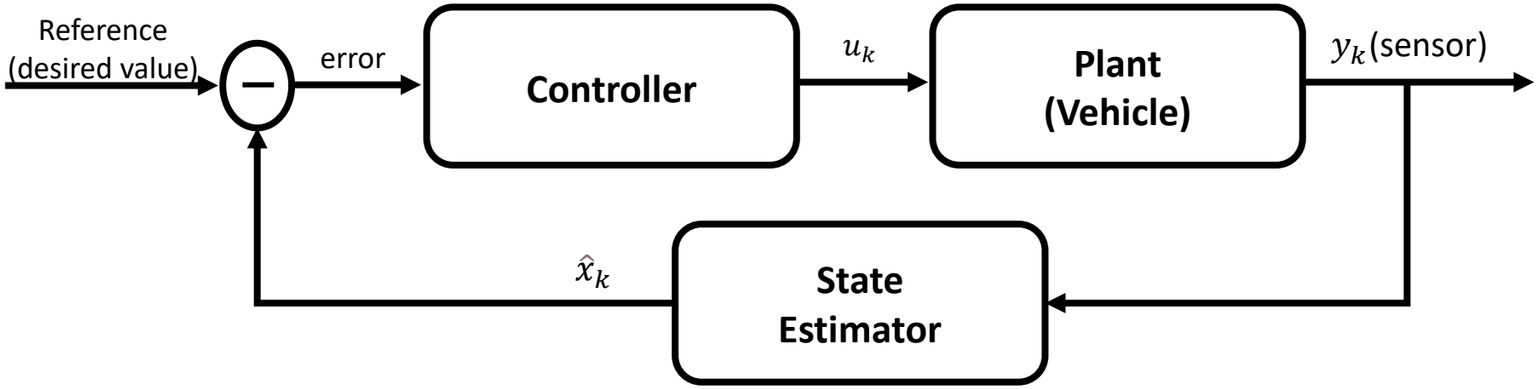
https://www.youtube.com/watch?v=1nJ79wX5EDM&ab_channel=%EB%A9%8D%EC%87%BC%EC%B8%A0

Beyond PID controller

- State estimator
  - Proper filter required!



노이즈를 가정하면
State Estimator (칼만필터)
필수!

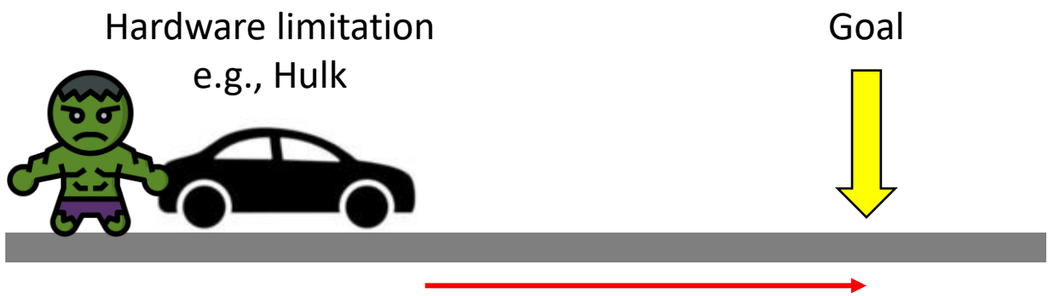


Beyond PID controller

2제어 쓸 때 주의할 것.

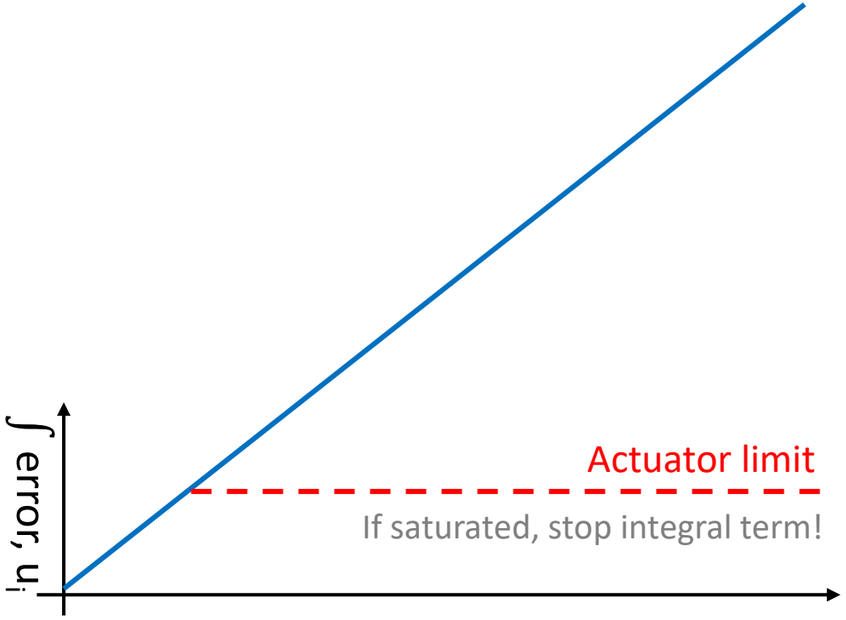
- Integrator anti-windup
  - Error 적분이 해소될 수 없을 때 적분 term 은 점점 커짐

<windup>



만약 이런식으로 차가 움직이면
에러누적 大. → windup

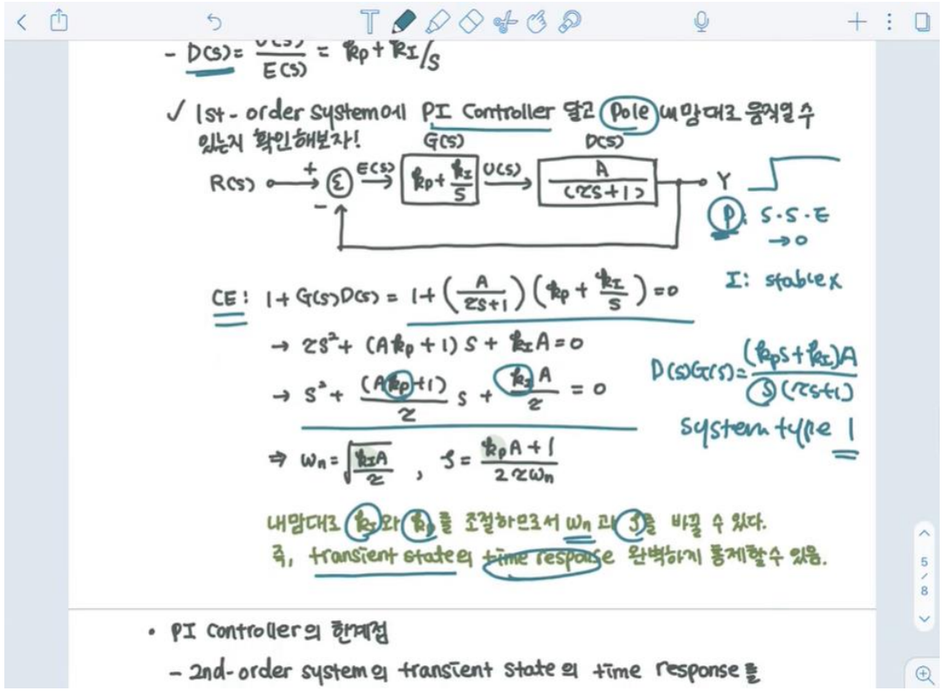
error의 최대값
지정해놓게
anti-windup



$$\delta_{pd} = K_p e + K_d \frac{d}{dt}(e) + K_i \int (e) dt$$

Beyond PID controller

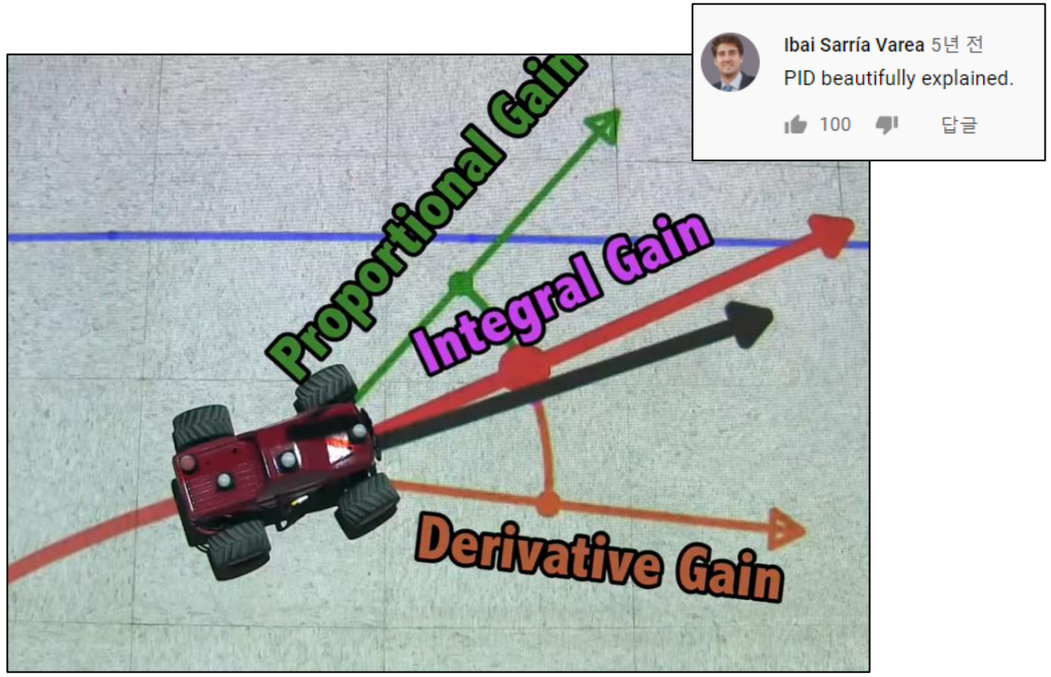
- Control engineering
  - 제어공학에서는 Plant와 PID controller가 수식으로 주어지면 input에 대한 system의 response를 계산하는 방법도 배웁니다
  - Laplace transform을 아는 것이 필수
  - 한국어 lecture “제어공학 뽀개기 (99%의 확률로 내가 부서짐)”이 유튜브에 공개되어 있으니 제어공학이 궁금하신 분들은 한번 들어 보시는 것도 좋겠습니다.
  - https://youtu.be/pVjKo_OVhU4 (강추)



Beyond PID controller

- Controlling Self Driving Cars (Video)
  - Aerospace Controls Lab (MIT)

    https://youtu.be/4Y7zG48uHRo



HL Klemove

Thank You